Inverse Elastic Shell Design with Contact and Friction

Mickaël Ly, Romain Casati, Florence Bertails-Descoubes, Mélina Skouras, Laurence Boissieux

Inverse Elastic Shell Design with Contact and Friction

MICKAËL LY, ROMAIN CASATI, FLORENCE BERTAILS-DESCOUBES,
MÉLINA SKOURAS, and LAURENCE BOISSIEUX,
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, France

(a) Target shape (b) Without inversion, sagging (c) Simulation after inversion, under wind motion

Fig. 1. Inverse design of an arched book page modeled by an artist. The target shape (a) is a half-cylinder which lies in contact with other pages on its two long edges. When simulated without prior inversion (i.e., simply initializing the natural shape with the target), the target shape (a) slides over the rest of the book (b). Thanks to our inverse process which simultaneously accounts for elasticity, gravity, and frictional contact (here with $\mu = 0.4$), we manage to recover a natural rest shape such that the target shape (a) is at stable equilibrium under surrounding forces, even for soft material parameters. The page can then be further simulated in a consistent manner, for instance under wind motion (c).

We propose an inverse strategy for modeling thin elastic shells physically, just from the observation of their geometry. Our algorithm takes as input an arbitrary target mesh, and interprets this configuration automatically as a stable equilibrium of a shell simulator under gravity and frictional contact constraints with a given external object. Unknowns are the natural shape of the shell (i.e., its shape without external forces) and the frictional contact forces at play, while the material properties (mass density, stiffness, friction coefficients) can be freely chosen by the user. Such an inverse problem formulates as an ill-posed nonlinear system subject to conical constraints. To select and compute a plausible solution, our inverse solver proceeds in two steps. In a first step, contacts are reduced to frictionless bilateral constraints and a natural shape is retrieved using the adjoint method. The second step uses this result as an initial guess and adjusts each bilateral force so that it projects onto the admissible Coulomb friction cone, while preserving global equilibrium. To better guide minimization towards the target, these two steps are applied iteratively using a degressive regularization of the shell energy.

We validate our approach on simulated examples with reference material parameters, and show that our method still converges well for material parameters lying within a reasonable range around the reference, and even in the case of arbitrary meshes that are not issued from a simulation. We finally demonstrate practical inversion results on complex shell geometries freely modeled by an artist or automatically captured from real objects, such as posed garments or soft accessories.

Authors' address: Mickaël Ly, mickael.ly@inria.fr; Romain Casati, romain.casati@gmail.com; Florence Bertails-Descoubes, florence.descoubes@inria.fr; Mélina Skouras, melina.skouras@inria.fr; Laurence Boissieux, laurence.boissieux@inria.fr, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000, Grenoble, France.

0730-0301/2018/11-ART201 $15.00
https://doi.org/10.1145/3272127.3275036

CCS Concepts: • **Computing methodologies** → **Animation**; **Physical simulation**;

Additional Key Words and Phrases: inverse design, dry frictional contact, thin elastic shell, cloth modeling and simulation

ACM Reference Format:
Mickaël Ly, Romain Casati, Florence Bertails-Descoubes, Mélina Skouras, and Laurence Boissieux. 2018. Inverse Elastic Shell Design with Contact and Friction. *ACM Trans. Graph.* 37, 6, Article 201 (November 2018), 16 pages. https://doi.org/10.1145/3272127.3275036

1 INTRODUCTION

From fluttering tree leaves to rolled paper and posed garments, surface objects are widespread in our environment. These thin structures are remarkable as they often exhibit intricate shapes, made of folds and wrinkles, while featuring complex motions, characterized by instabilities such as buckling. Ingredients responsible for such a visual richness include internal elasticity of the structures, with a preference for *bending*, combined with a possibly curved *natural shape*, and finally, the contribution of gravity, contact and friction. The theory of *thin elastic shells*, coupled to a frictional contact model, is a natural way to model all these features.

Modeling freely shell surfaces, or capturing the geometry of real shells, are two research topics that are still very active, see [Li et al. 2017; Pons-Moll et al. 2017]. Surfaces resulting from such reconstruction processes naturally encode all the surrounding forces applied on the shell, such as internal elasticity, gravity, and frictional contact. However, there is currently no way of extracting such mechanical information from the onset of a mere surface. Thus, there is no possibility to further simulate these objects in a consistent manner. Simply injecting the modeled surface as the natural shape of a shell simulator irremediably results in a sagging and/or slipping of the structure when one applies gravity, thus losing the original intended shape (see Figure 1).

In this paper we explore a new strategy for modeling thin elastic shells both realistically and intuitively, by combining the fine user control offered by geometric tools together with the predictive capabilities of simulation. Our idea is to devise an automatic *inverse modeling* process able to interpret a merely geometric surface as a stable equilibrium state of a thin shell mechanical simulator. Leaving the choice to the user concerning the material properties of the shell (mass density, stiffness, friction coefficients), our algorithm takes as input a target (deformed) 3D surface, resulting from either geometric design or capture. Our method then automatically retrieves a plausible natural shape for the shell as well as frictional contact forces at play. When further simulated with the same parameters under external forces, the shell perfectly matches the input pose, remains still, and may be subsequently animated under new external forces.

To the best of our knowledge, our method is the first able to invert exactly *any* arbitrary shape subject to *contact* and *dry (Coulomb) friction*. In particular, our algorithm can handle large displacements between the input target shape and the output natural shape, and is able to accommodate both local contraction and dilatation of the shell during inversion. As a result, our method can be used to invert a wide range of shell surfaces, either stemming from realistic objects such as real garments (see Figure 9), or from free form design, such as the stylized hat represented in Figure 2. Our method can even accommodate fanciful designs such as cartoonish garments that would be tricky to actually handcraft (see Figure 8).

(a) Target (b) Sagging (c) Simulation after inversion

Fig. 2. Breathing physics into a floppy hat designed in 3D by an artist (a). Without inversion, the hat sags under gravity, loosing its intended style and partially covering the face of the character (b). In contrast, our method automatically infers a natural shape of a thin elastic shell simulator such that a stable equilibrium matches the input geometry (a). Starting from this input pose, the hat can then be animated in a convincing manner as the character shakes her head, while preserving its original style (c).

2 RELATED WORK

Inverse problems have been studied in a huge amount of fields ranging from mechanical engineering to meteorology, electro-magnetism and more recently biology. The basic principle is to retrieve unknown parameters of a model from some observations of the object of interest. In mechanics, inverse problems can be classified into two main categories [Beck and Woodbury 1998]: (a) inverse *measurement* problems, where material properties (e.g., mass density, stiffness, internal damping) are looked for; and (b) inverse *design* problems, where the undeformed configuration of the object is sought for. Our problem clearly falls into category (b): the user provides all material parameters (including friction coefficients), and our goal is to find a natural shape for the shell which ensures stable equilibrium of the target shape for these particular material parameters.

The computer graphics community has focused much attention in inverse design problems during the two last decades, motivated by various applications ranging from animation authoring to soft object fabrication. We review here the most relevant references, with an emphasis on *cloth inverse design*, which is currently the line of research that is closest to our problem.

2.1 Inverse elastic design

In computer graphics, the first methods dedicated to inverse design were introduced in the context of inextensible thin elastic rods [Derouet-Jourdan et al. 2010; Hadap 2006], with the aim of facilitating hair posing. These approaches leveraged reduced (angle or curvature based) coordinates models for rods to simplify the inverse static problem, which boils down to a linear equation in the case when only gravity and bending/twisting elasticity are accounted for. Derouet-Jourdan et al. [2010] have even provided a sufficient condition for stability that only depends on the material parameters of the rod. Later, they have extended the equilibrium condition to account for frictional contact in a rod assembly [Derouet-Jourdan et al. 2013]. They have shown that under reasonable assumptions on the natural shape, the full inverse static problem could be cast in a form that is very close to the one-step direct dynamic problem for rods, and could thus be solved using efficient solvers of the literature [Daviet et al. 2011] (albeit losing the guarantee of stability).

Unfortunately, thin elastic shells, which do not only bend but also stretch, do not, to our knowledge, benefit from such a reduced formulation. The simplified inverse formulations mentioned above are thus not directly transferable to shells. As the vast majority of existing shell simulators favor a nodal coordinate representation, and in the absence of a more suitable parameterization, we have also opted for a nodal model. This default choice however implies that we have to cope with a strongly nonlinear inverse static problem, even in the absence of contact and friction.

It turns out that a number of works have looked at inverse design problems for nodal systems. Twigg and Kačić-Alesić [2011] have investigated the inverse static problem for mass-spring systems, solving for unknown spring rest lengths. Their method attempts to minimize the sum of active forces, but unfortunately does not necessarily bring it to a vanishing point, thus failing to guarantee equilibrium. Considering the more general settings of finite elements (FEM), several works have considered solving nonlinear inverse static problems, using different strategies. For elastic membranes subject to pressure forces, Skouras et al. [2012] have proposed a technique to optimize the membranes' deflated shapes based on an augmented Lagrangian method capable of inverting large nonlinear stretch. In a subsequent work [Skouras et al. 2014], they have focused on inflatables made of quasi-inextensible materials that exhibit a large resistance to stretching while being compliant to bending — materials that share some similarities with the materials we use in this work. However, the relaxed formulation that they have use to handle compression, which works well in the case of problems dominated by in-plane deformations, is not directly exploitable in another context. In the case of 3D elastic objects subject to gravity, Chen et al. [2014] have shown that the asymptotic numerical model was a more robust and efficient solving method compared to the classical Newton-Raphson method.

The approaches described above were efficient and robust enough to handle many examples that could eventually be fabricated and validated in real. However, none of them has ever accounted for scenarios involving contact and dry friction forces, which complexifies even more the inverse static problem and even changes its nature, transforming a nonlinear equality into a nonlinear *inclusion* in a convex set. Furthermore, even in the absence of frictional contact, inverting a soft membrane subject to gravity is particularly challenging due to the large displacements that occur between the natural shape and its shape at equilibrium. We have tried in the past several inverse algorithms that were said to be successful in other contexts, for instance in the case of soft objects modeled with co-rotated linear FEM [Wang et al. 2015], but they have all failed when applied to our problem. We have thus striven to design a particularly *robust* inversion algorithm, able to account not only for gravity applied onto a soft thin shell, but also for *contact* and *dry friction* with an external body.

Closer to our challenges, research on *cloth* modeling has recently been focused on new inverse design methods, through a series of papers dedicated to traditional garments made of flat patterns. We review these methods below.

2.2 Cloth inverse design

Physics-based pattern adjustement. The real process for making wearable garments involves the design of 2D patterns, i.e., flat patches which are then manufactured with real fabric and sewn together to create the final garment. Early cloth simulation methods already mimicked this process in order to dress virtual characters before animating them [Carignan et al. 1992]. To improve the garment modeling process, Volino and colleagues [2005] have later imagined an intuitive and interactive design environment, where the user can simultaneously edit 2D patterns and visualize the 3D result under gravity and body contacts thanks to a fast draping simulator. Such an interactive physics-based tailoring process is now the standard work flow used in many commercial tools like the popular Marvelous designer software [2010]. It has also inspired further work in academia [Umetani et al. 2011], which has improved the 3D responsiveness to 2D pattern editing and thus greatly smoothened the creative flow.

Recently, Bartle and colleagues [2016] have allowed the user to edit the garment directly in the 3D space, through cutting, lengthening, or merging operations. In their approach, the new 3D target shape is closely put in correspondence with the result of the static cloth simulator by automatically adjusting the corresponding 2D patterns. This adjustment is performed through a gradient-free, fixed point procedure, which locally optimizes the size of each mesh triangle. At each iteration, the resulting set of disconnected triangles is embedded in a 2D mesh with minimal distortion. This algorithm is fast enough for allowing interactive adjustment of a given garment. However, it converges only if the natural shape of the garment results from a contraction of the target. Although this assumption is reasonable in the case of a detailed garment pose, which would exhibit all the realistic wrinkles even at fine scales, it is too restrictive when dealing with an arbitrary surface coming from 3D design or automatic capture. As shown in Figure 10, modeling a puff sleeve as a stylized roundish shape leads to the retrieval of an inflated natural pose – without allowing for inflation, the corresponding inverse problem would have no solution. Another limitation of Bartle et al.'s approach is its difficulty to include details to an existing garment such as new fold lines located at a specific place. To incorporate new folds, Li et al. [2018b] have lately abandoned the idea of reaching a precise target geometry. Instead, they rely on user strokes to guide the physical pattern making process and reach a shape that resembles the intended design, in the limits of the chosen material parameters. In contrast, our method can take as input any smooth mesh with arbitrary curvature pattern. It is moreover totally agnostic of any garment template, and only requires a 3D mesh as input. Very recently, Wang [2018] has in turn relaxed the constraint of achieving a precise 3D target and has instead favored fitting-to-new-body criteria for improving actual manufacturing of garments of different sizes, given a template — a goal fundamentally different from ours. Like us however, the author is faced with high precision needs. For this reason, his method also relies on a gradient-based optimization scheme, which is made very efficient thanks to careful implementation choices. However, friction is not taken into account in this approach.

Free form cloth modeling. An interesting alternative in cloth modeling is to leverage 3D geometric tools for directly sculpting the final pose regardless of the underlying physical process at the origin of the cloth deformation. Two families of geometric methods are currently available: manual geometric design, and automatic geometric acquisition. On the one hand, advanced shape editing tools allow 3D designers to carve the drape of a garment directly around a virtual character (see, e.g., [Porumbescu et al. 2005]). For fairly simple garments, sketch-based interfaces may greatly improve the intuitiveness and speed of the process [Igarashi and Hughes 2003; Li et al. 2017; Turquin et al. 2007]. Once a final garment has been created, some geometric transfer methods can be used for automatically adapting it to various character morphologies, while preserving the style [Brouet et al. 2012]. On the other hand, thanks to the considerable advance of image-based capture these latest years, it becomes now affordable to acquire precisely the full 3d geometry of static [Bradley et al. 2008; White et al. 2007] and even dynamic [Pons-Moll et al. 2017] cloth with folds and wrinkles, thus allowing for an automatic garment creation from a real source.

Geometry-based techniques are appealing because they provide the user with full control over the final 3D shape of the garment. Realistic but also imaginary garments (see Figure 8) can be created this way, leaving aside the flat patch-based structure of garments and focusing instead on the free-form 3D appearance. Yet, since the modeling process is done independently from physical simulation, the resulting geometry cannot be interpreted mechanically in a straightforward way. This implies that the modeled garment cannot be animated nor physically modified easily. A naive approach consists in plugging the resulting shape as the rest (undeformed) configuration of a shell simulator. However, once the simulation starts, the shape irremediably sags under gravity and may greatly diverge from the desired pose, thus ruining all prior modeling efforts (see Figure 2).

2.3 Our contribution: a physical interpretation of surfaces

Our goal here is not to help fabricate real garments using the standard pattern-based fabrication process. Instead, we intend to interpret any arbitrary input surface as a physically deformed shell subject to internal elasticity, gravity, contact and friction. Unlike recent inverse cloth design methods, we maintain the matching of a precise 3D shape as our primary goal. We thus depart from the representation of natural surfaces as flat patches, which is standard in garment modeling, and instead considerably enlarge our solution space, now embedded in the 3D space. Moreover, our method imposes no restriction *a priori* on the type or magnitude of inverse deformation that is necessary to interpret the target shape. To tackle the complexity of such an inverse problem, which is exacerbated when contact and friction enter the game, we rely on a gradient-based optimization approach, which incorporates nonsmooth terms due to dry frictional contact. To the best of our knowledge, this is the first time inverse design is successfully performed on a nodal system with contact and friction. In a prepublication, a first approach towards this goal was presented, but failed to converge robustly in a vast majority of cases [Casati et al. 2016].

Our new strategy, which interleaves two successive steps — one approximating contacts as fixed points, the other adjusting contact forces so that they fulfill frictional contact constraints — allows us to invert a wide range of target shapes with a high precision. Although of course not as fast as pattern-adjustment methods, our computational timings still remain tractable. For instance, for a mesh composed of 900 vertices and subject to 200 contact points, our algorithm takes at most a few hours to converge to a high precision, on a standard PC and with a CPU implementation.

3 ALGORITHM OVERVIEW

We propose an original inverse method for interpreting an arbitrary surface geometry as the static equilibrium configuration of a thin elastic shell deformed by gravity and frictional contact with an external body.

In our approach, the natural shape of the garment is left as an unknown of the problem, and (unknown) frictional contact forces are considered to account for the input pose. Material parameters (mass, stiffness, friction coefficient) are, in contrast, not treated as unknowns and can be freely chosen by the user. For the sake of simplicity, we consider the discrete shell model of Grinspun et al. [2003], but our inverse approach could be used with other shell energy formulations, including finite element models. Frictional contact is in turn modeled using the Signorini-Coulomb law, which states that at static equilibrium, each contact force should lie in the Coulomb cone, oriented outwards in the local contact basis. For direct simulations, we use the nodal frictional contact simulator of [Daviet et al. 2015; Li et al. 2018a], with fixed resolution. Note that our method is tight to the Signorini-Coulomb model, but not to a particular frictional contact solver: any solver satisfying non-penetration and Coulomb constraints could be employed to simulate shapes inverted by our approach in a consistent manner. The mechanical models we use along with notation are detailed in Section 4.

We assume that the target shape is represented by a triangular mesh with n vertices whose stacked positions are denoted by $x_\mathrm{t} \in \mathbb{R}^{3n}$. We discretize the mid-surface of the shell in its natural and deformed configurations using meshes with the same connectivity, whose positions are denoted by $\bar{x} \in \mathbb{R}^{3n}$, respectively $x \in \mathbb{R}^{3n}$. For the sake of simplicity, we assume that our meshes are sufficiently refined so that we can consider contacts at mesh vertices only.

In the presence of frictional contact, the static equilibrium of the shell is given by

$$F(x, \bar{x}) = f_c, \tag{1}$$

where F represents the total nodal conservative forces acting on each point of the shell and $f_c \in \mathbb{R}^{3n}$ are the nodal frictional contact forces. As we will see in Section 4, unlike conservative forces, these frictional contact forces have no explicit expression, but they should satisfy certain conical constraints involving a convex set K (corresponding to a product of cones) such that

$$f_c \in K. \tag{2}$$

The inverse problem to solve can then be formulated as

$$\text{find } \bar{x} \quad \text{such that} \quad F(x_\mathrm{t}, \bar{x}) \in K. \tag{3}$$

This is a nonlinear inclusion in a convex set, which may have no solution, or multiple solutions, therefore defining an *ill-posed* problem. It also involves contact force unknowns, which makes the problem particularly challenging. To support noisy input target shapes, we do no try to solve this problem exactly but rather aim to solve the more robust least-squares formulation

$$\min_{\substack{(x,\bar{x}) \\ F(x,\bar{x})\in K}} \frac{1}{2}\|x - x_\mathrm{t}\|^2, \tag{4}$$

with $\| \bullet \|$ the Euclidean norm defined on $\mathbb{R}^{3n}$.

However, under this form, our problem is complex to handle numerically because of the particular form of the constraint (nonlinear inclusion in a product of cones). Instead, we retrieve a nonlinear equation for the constraint,

$$\min_{\substack{(x,\bar{x}) \\ G_K(F(x,\bar{x}))=0}} \frac{1}{2}\|x - x_\mathrm{t}\|^2, \tag{5}$$

by introducing the new function G_K (whose exact definition will be given in Section 6), which satisfies

$$y \in K \iff G_K(y) = 0. \tag{6}$$

Solving Problem (5) directly is still challenging due to the high nonlinearity of the constraint. Therefore, we devise a two-step algorithm that proceeds as follows.

In the first step (Step 1), described in Section 5, we assume that all points in contact are fixed and we solve the more tractable unconstrained problem

$$\text{find } \bar{x}_{\frac{1}{2}} \quad \text{such that} \quad \bar{x}_{\frac{1}{2}} = \operatorname*{argmin}_{\bar{x}} \frac{1}{2}\|\Phi(\bar{x}) - x_\mathrm{t}\|^2, \tag{7}$$

where $\Phi : \mathbb{R}^{3n} \to \mathbb{R}^{3n}$ is the so-called *draping* function, that maps the natural pose of the shell to a stable equilibrium configuration, i.e., $\Phi(\bar{x}) = x$. This gives us a first estimate $\bar{x}_{\frac{1}{2}}$ of the natural shape and its corresponding draped shape $x_{\frac{1}{2}} = \Phi(\bar{x}_{\frac{1}{2}})$, whose contact-free points properly satisfy force balance as well as stability of the equilibrium.

In the second step (Step 2), provided in Section 6, we aim at projecting $(x_{\frac{1}{2}}, \bar{x}_{\frac{1}{2}})$ onto the constraint manifold $G_K \circ F = 0$, so as to rectify possible Coulomb conical constraints violation at contact points, while maintaining the contact-free points at equilibrium. This projection problem is however extremely challenging. We have found in practice that solving instead the (simpler) following minimization problem,

$$\min_{\bar{x}} \frac{1}{2} \|G_K(F(x_{\frac{1}{2}}, \bar{x}))\|^2, \tag{8}$$

always brings back our solution to the constraint manifold with a very high precision.

Finally, as we shall see in Section 5, a given natural shape may map to several distinct draped shapes. This behavior may cause trouble to the gradient-based minimizers that we use, which might get stuck into local minima or even fail. To guide the algorithm towards the draped pose closest to the target, we augment the potential energy of the shell with a regularization term of the form $\frac{\lambda}{2}\|x - x_t\|^2$. We then iterate the two steps of our algorithm using a sequence of λ with decreasing values. Thus, our final algorithm, summarized by Algorithm 1, acts like an homotopy (or continuation) method that progressively and alternately refines the shapes corresponding to the contact-free and contact zones of the shell by solving problems of increasing difficulty.

ALGORITHM 1: Robust inversion algorithm.

Data: Target equilibrium pose x_t, initial regularization factor λ_0, regularization reduction factor $\alpha \in]0\,;1[$
Result: A pair $(x, \bar{x})$ consisting of a stable equilibrium pose and a natural pose with x as close as possible to x_t

```
// No a priori knowledge of the natural shape
k ← 0 ;
x̄_0 ← x_t ;
while λ_k > 0 do
    if λ_k ≤ ε_λ then
        // Last iteration with no regularization
        λ_k ← 0;
    end
    // Step 1  (EvalObjective is defined in Algorithm 2 of Section 5)
    (x_{k+1/2}, x̄_{k+1/2}) ← BFGS_min(EvalObjective(x_t, λ_k, •),
      x̄_init = x̄_k) ;
    // Step 2  (G_K is defined in Section 6)
    (x_{k+1}, x̄_{k+1}) ← BFGS_min (G_K(F(x_{k+1/2}, •)), x̄_init = x̄_{k+1/2});
    // Decrease λ
    λ_{k+1} ← α λ_k ;
    k ← k + 1 ;
end
```

4 MECHANICS

Notation. If $F(x, \bar{x})$ is a vector-valued function defined on $\mathbb{R}^{3n} \times \mathbb{R}^{3n}$, then $\mathrm{D}_x F(x, \bar{x})$ and $\mathrm{D}_{\bar{x}} F(x, \bar{x})$ denote its Jacobian w.r.t. the first and second variable respectively. For a single variable function $\Phi(\bar{x})$, we shall omit the subscript in the Jacobian, that is writing $\mathrm{D}\Phi(\bar{x})$. For a real-valued function $f(x, \bar{x})$, we shall use the gradient notation $\nabla_x f(x, \bar{x})$ and $\nabla_{\bar{x}} f(x, \bar{x})$ instead, with $\nabla_x f(x, \bar{x}) = (\mathrm{D}_x f(x, \bar{x}))^\top \in \mathbb{R}^{3n}$. $\|x\|$ will denote the L_2 norm of x. Scalar product between two vectors x and y will be denoted by $x^\top y$, with $x^\top$ the transpose of x.

4.1 Shell model

We model our thin shell using the simple discrete shell model proposed by Grinspun et al. [2003], which enjoys widespread popularity in computer graphics. However, in the definition of the membrane energy, we only keep the contribution of edge elasticity, as the contribution of face elasticity turns out to be redundant in the case of a triangular mesh. Given a deformed shell with nodal positions x and a natural configuration $\bar{x}$, the shell internal energy then reads

$$E_{\text{int}}(x, \bar{x}) = E_{\text{membrane}}(x, \bar{x}) + E_{\text{bending}}(x, \bar{x})$$

with the simplified membrane energy

$$E_{\text{membrane}}(x, \bar{x}) = \frac{k_L}{2} \sum_{e \in \mathcal{E}} \frac{\left(L_e - \bar{L}_e\right)^2}{\bar{L}_e} \tag{9}$$

and the bending energy

$$E_{\text{bending}}(x, \bar{x}) = \frac{k_\theta}{2} \sum_{e \in \mathcal{E}_{int}} \frac{3\bar{L}_e^2}{\bar{A}_e} \left(\theta_e - \bar{\theta}_e\right)^2, \tag{10}$$

where $\mathcal{E}$, resp. $\mathcal{E}_{int}$, denote the sets of all, resp. internal, edges.

The quantities L_e and θ_e represent the deformed per-edge length and per-edge dihedral angle respectively, which can all be computed from the deformed configuration x. Their natural counterparts are denoted by the barred quantities $\bar{L}_e$ and $\bar{\theta}_e$, which can similarly be computed from $\bar{x}$ (for the sake of simplicity we have omitted writing explicitly their dependence on $\bar{x}$). A_e is the sum of the areas of the faces adjacent to edge e. The constant k_L represents in-plane stiffness and k_θ the bending stiffness.

Note that the mechanical behavior of this edge-based model is tied to the resolution of the mesh: for the same material parameters, its in-plane behavior stiffens when the resolution increases. Therefore, these parameters cannot be directly compared to bulk properties like the Young's modulus. However, we could easily replace E_{int} by another formulation, e.g., compatible with a finite element discretization.

4.2 Gravitational energy

Let ρ denote the area density of the shell. The mass m_i of the node i can be computed by lumping face masses at vertices as

$$m_i(\bar{x}) = \sum_{f \in \mathcal{T}_i} \frac{1}{3} \rho \bar{A}_f, \tag{11}$$

where $\mathcal{T}_i$ is the set of triangles incident to vertex i.

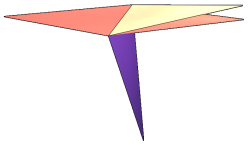
Letting g denote the constant of gravity, and e_z be the unitary upward vector, the gravitational energy can then be written as

$$E_g(x, \bar{x}) = \sum_{i \in \mathcal{V}} m_i(\bar{x})\, g\, x_i^\top e_z, \tag{12}$$

where $\mathcal{V}$ is the set of all vertices.

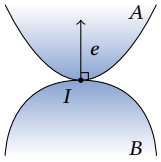
We draw the reader's attention to the fact that the nodal masses do depend on the natural shape. At first sight, it might be tempting to use pre-computed masses based on the target mesh geometry and keep them constant during the inversion. However, as we are using

an area density and as the shell might locally stretch or compress, using the target geometry directly may lead to a wrong mass allocation. One consequence is illustrated in the inset figure, which depicts an initially flat shell made of two triangular faces (in orange), whose left face is fixed. Due to gravity, the free face of the corresponding deformed mesh (in purple) is stretched when at equilibrium. Using masses computed from this deformed shape results in an overestimation of the weight of the hanging node, which can only be compensated by an extra bending of the natural shape (in yellow). In contrast, when correctly setting the dependence of the mass to the unknown natural shape (and of course differentiating energies accordingly), the original flat shape (in orange) is exactly[1] retrieved.



4.3 Frictional contact model

We model contact and friction using the Signorini-Coulomb model, which offers a good compromise between simplicity and realism. We briefly summarize this model below, and we refer to the book by Acary and Brogliato [2008] for a comprehensive exposition. Let A and B be two bodies in contact at point I at a certain instant in time. We assume that the boundary of these objects is smooth in the vicinity of the contact point so that we can define a normal vector $e \in \mathbb{R}^3$ at I pointing from B to A, as illustrated in the inset figure. We let $r \in \mathbb{R}^3$ denote the force exerted by B on A at I, and $u \in \mathbb{R}^3$ the relative velocity of A with respect to B at I. We assume that we are given a coefficient of friction $\mu \geq 0$.



The Coulomb friction cone $C(e, \mu)$ is defined as

$$C(e, \mu) = \{x \in \mathbb{R}^3 / \|x_T\| \leq \mu x_N\}, \tag{13}$$

where $x_N = x^\top \frac{e}{\|e\|} \in \mathbb{R}$ and $x_T = x - x_N \frac{e}{\|e\|} \in \mathbb{R}^3$ are, respectively, the normal and tangent components of x with respect to e.

The Signorini-Coulomb model classifies frictional contact into three distinct configurations for the vectors u and r, as depicted in Figure 3,

- taking-off: $r = 0$ and $u_N > 0$
- sticking: $r \in \operatorname{Int} C(e, \mu)$ and $u = 0$
- sliding: $r \in \partial C(e, \mu)$, $r \neq 0$, $u_N = 0$ and $u_T = -\alpha r_T$ $(\alpha > 0)$,

where Int$C$ denotes the interior and ∂C the boundary of the set C.

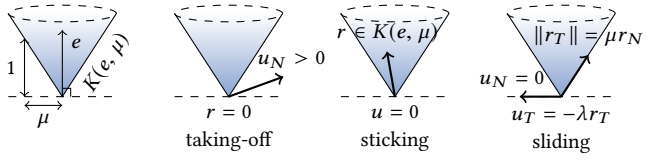


Fig. 3. The friction cone $C(e, \mu)$ and the three configurations of the Signorini-Coulomb law.

[1]We exactly retrieve the original natural shape here because the test case considered (bitriangle) is elementary. In the general case, multiple natural shapes may be retrieved even when computing the mass consistently, due to the ill-posedness of our problem.

4.4 Static equilibrium formulation

Let us first consider that only conservative forces act on our shell model. These forces, denoted by F, derive from the total potential energy, which includes internal elasticity and gravity,

$$E_p(x, \bar{x}) = E_{\text{int}}(x, \bar{x}) + E_{\text{g}}(x, \bar{x}),$$

such that $F(x, \bar{x}) = -\nabla_x E_p(x, \bar{x})$. Note that other external energies, such as potential fields (e.g., for modeling penalty forces), could also be considered in this formulation.

In the absence of contact, the static equilibrium of the shell is characterized by total nodal force balance, i.e.,

$$F(x, \bar{x}) = 0. \tag{14}$$

A *stable* static equilibrium state is in turn guaranteed if the Hessian $\nabla^2_x E_p(x, \bar{x}) = -\mathrm{D}_x F(x, \bar{x})$ is positive.

Let us now assume that some vertices of the deformed mesh, whose set of indices is denoted by I_c, are in contact with an external object. The static equilibrium of the shell implies that all its nodes have zero velocity, which in turn means that the system is the sticking configuration. Consequently, if $i \in I_c$, then f_{c_i}, the contact force exerted on the vertex i, must belong to the interior of the Coulomb friction cone $C(e_i, \mu_i)$ where e_i and μ_i denote the local mesh normal and the local friction coefficient respectively. If $i \notin I_c$, then the point is at equilibrium without contact, meaning that f_{c_i} must vanish.

By noting

$$K = \prod_{i=1}^{n_v} K_i \quad \text{with} \quad K_i = \begin{cases} \operatorname{Int} C(e_i, \mu_i) & \text{if } i \in I_c \\ \{0\} & \text{otherwise,} \end{cases} \tag{15}$$

the static equilibrium of the shell in the presence of contact can then be written as the inclusion

$$F(x, \bar{x}) \in K. \tag{16}$$

5 INVERSION WITHOUT CONTACT CONSTRAINTS

The purpose of the first step of our algorithm is to compute the natural shapes, as well as the corresponding deformed shapes, of the contact-free zones of the shell. We assume for now that no point is constrained and that we are optimizing the entire mesh. This case will then be easily generalized to the inclusion of fixed vertices.

Our goal is to find a natural mesh $\bar{x}_{\frac{1}{2}}$ corresponding to a deformed mesh which is as close as possible to the target x_{t} while being in static equilibrium. We would also like this equilibrium to be *stable*. That is, the shell should go back to its initial deformed configuration when slightly moved apart from it. Formally, we aim to solve the constrained minimization problem

$$\min_{\substack{(x, \bar{x}) \\ F(x, \bar{x}) = 0 \\ \nabla^2_x E_p(x, \bar{x}) > 0}} \frac{1}{2} \|x - x_{\text{t}}\|^2. \tag{17}$$

Because of the high nonlinearity of the internal energy E_p, we anticipate here the fact that inverse statics will be a difficult problem, *even in the absence of contact and friction*. The stability condition, in particular, is challenging to satisfy in practice because it involves the computation of the eigenvalues of the Hessian $\nabla^2_x E_p(x, \bar{x})$, which is computer-expensive. To circumvent this difficulty, we model our

full inverse problem — including the stability condition — using an energy-based formulation, i.e., for an optimal $\bar{x}_{\frac{1}{2}}$ parameter to be found, the optimized deformed pose $x_{\frac{1}{2}}$ should match a local minimum of the function $x \mapsto E_p(x, \bar{x}_{\frac{1}{2}})$. That is, $x_{\frac{1}{2}}$ should correspond to a *draped pose* of the shell simulator initialized with $\bar{x}_{\frac{1}{2}}$ as the natural pose.

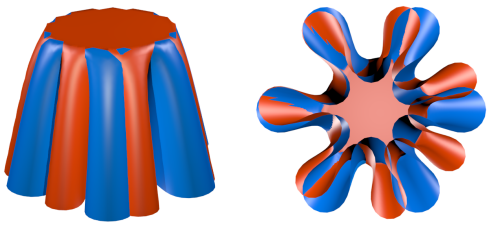
Thus, our approach departs from the more standard strategy that consists in directly minimizing the norm of the total forces acting on the system, i.e., solving $\min_{\bar{x}} \frac{1}{2}\|F(x_t, \bar{x})\|^2$, which we had also experimented initially. While simpler at first sight, this technique is not robust as it enforces the search for a vanishing point $F(x, \bar{x})$ *strictly along the* $x = x_t$ *subspace*, even if it means finding a $\bar{x}_{\frac{1}{2}}$ that does not guarantee equilibrium at $(x_t, \bar{x}_{\frac{1}{2}})$.

5.1 Least-squares formulation using the draping function

We detail below the algorithm that we use to solve Problem (17) using the draping function.

Implicit draping function. Let $\bar{x}$ be a natural pose (non necessarily optimal). In the case when $x \mapsto E_p(x, \bar{x})$ is a strictly convex function, then a unique draped pose x exists, corresponding to the (global) minimum of the energy. We can thus define a *draping function* on $\mathbb{R}^{3n}$ as $\Phi : \bar{x} \mapsto \operatorname{argmin}_x E_p(x, \bar{x})$.

In the general case however, the potential energy is not convex. It may feature several local minima, thus precluding the definition of such a unique mapping on $\mathbb{R}^{3n}$. This is illustrated in Figure 4.



(a) Side view (b) Bottom view

Fig. 4. For this tablecloth, at least two different draped poses, one in blue and one in orange, may be obtained from the same flat natural pose $\bar{x}$ by perturbing the initial state.

Yet, in the nonconvex case a draping function Φ can still be defined *locally*. Let us consider a particular natural pose $\bar{x}^*$ (again, non necessarily optimal), and a deformed pose x^* such that x^* is a strict local minimum of $x \mapsto E_p(x, \bar{x}^*)$. We thus have $F(x^*, \bar{x}^*) = 0$ and $\mathrm{D}_x F(x^*, \bar{x}^*) \succ 0$. According to the implicit function theorem, there exists a unique mapping Φ from a neighborhood $\bar{V}$ of $\bar{x}^*$ to a neighborhood V of x^* such that $x = \Phi(\bar{x})$ with $\bar{x} \in \bar{V}$ and $x \in V$. Intuitively, in our case computing $\Phi(\bar{x})$ would consist in minimizing the potential energy $x \mapsto E_p(x, \bar{x})$ in the neighborhood of $(x^*, \bar{x}^*)$. Moreover, the theorem says that $\mathrm{D}_x F$ is invertible on $V \times \bar{V}$ and that Φ is differentiable on $\bar{V}$, with an explicit expression for its Jacobian,

$$\mathrm{D}\Phi(\bar{x}) = -\left(\mathrm{D}_x F(\Phi(\bar{x}), \bar{x})\right)^{-1} \left(\mathrm{D}_{\bar{x}} F(\Phi(\bar{x}), \bar{x})\right). \tag{18}$$

Least-squares minimization. Thanks to the existence of a local draping function Φ, our least squares inverse problem reads

$$\text{find } \bar{x}_{\frac{1}{2}} \text{ such that } \quad x_t = \Phi(\bar{x}_{\frac{1}{2}}), \tag{19}$$

and can be relaxed in the least squares sense as

$$\text{find } \bar{x}_{\frac{1}{2}} \text{ such that } \quad \bar{x}_{\frac{1}{2}} = \operatorname*{argmin}_{\bar{x}} \underbrace{\frac{1}{2}\|\Phi(\bar{x}) - x_t\|^2}_{J(\bar{x})}. \tag{20}$$

Problem (20) is more likely to possess a solution $\bar{x}_{\frac{1}{2}}$, which in turn is solution to (19) if and only if the objective function $J(\bar{x})$ vanishes at $\bar{x}_{\frac{1}{2}}$. However, if $J(\bar{x}_{\frac{1}{2}})$ does not exactly vanish we still get some very useful information. Indeed, solving (20) provides us with a deformed pose $\Phi(\bar{x}_{\frac{1}{2}})$ which is *guaranteed* to be a minimum configuration of the energy function $x \mapsto E_p(x, \bar{x}_{\frac{1}{2}})$. Moreover, this deformed pose is as close as possible to the target x_t.

Formulation (20) is common for inverse problems. To minimize J in practice, we use the BFGS approach , which relies on the computation of ∇J. A classical strategy for efficiently computing ∇J is the *adjoint* method [Giles and Pierce 2000], which is briefly described in Section 5.2. Algorithm 2 provides the pseudo-code for evaluating the objective function and its gradient based on the adjoint method, at each iteration of the inversion algorithm.

Yet, a major difficulty remains. Indeed, the adjoint method has the special feature of relying only on the evaluation of Φ to compute ∇J, and not that of $\mathrm{D}\Phi$. However, in our case we are faced with an awkward situation, as the draping function Φ is defined *locally*. In Section 5.3 we shall explain how to evaluate Φ in a consistent way during subsequent steps of the minimization.

ALGORITHM 2: Evaluation of the objective function J and of its gradient ∇J (under regularized energy)

Data: Target equilibrium pose x_t, energy regularization factor λ, current natural pose $\bar{x}$
Result: The value and the gradient of the objective function J at $\bar{x}$, with energy regularization factor λ
Procedure `EvalObjective`$(x_t, \lambda, \bar{x})$

```
// Draping is defined in Algorithm 3 of Section 5
x ← Draping(x_t, λ, x̄);
objective ← ‖x − x_t‖²;
p ← Linear_solver((D_x F(x, x̄))^⊤ , x − x_t);
gradient ← −(D_x̄ F(x, x̄))^⊤ p;
return x, objective, gradient;
```

5.2 The adjoint method

To solve the minimization problem (20) robustly, it is desirable to compute the gradient ∇J accurately. Differentiating the objective function J gives

$$\nabla J = \mathrm{D}\Phi(\bar{x})^\top \left(\Phi(\bar{x}) - x_t\right). \tag{21}$$

Recall that an explicit expression for $\mathrm{D}\Phi(\bar{x})$ is provided by (18). However, it is dense in general and its computation requires a full matrix inversion.

Actually, we do not really need to compute $\mathrm{D}\Phi(\bar{x})$, but only ∇J. By replacing $\mathrm{D}\Phi(\bar{x})$ with its expression (18), the adjoint method consists in decomposing the computation of ∇J into two steps,

$$\begin{cases} (\mathrm{D}_x F(\Phi(\bar{x}), \bar{x}))^\top \; p &= \Phi(\bar{x}) - x_\mathrm{t} \\ \nabla J(\bar{x}) &= -(\mathrm{D}_{\bar{x}} F(\Phi(\bar{x}), \bar{x}))^\top \; p, \end{cases} \tag{22}$$

where p is called the *adjoint state*. Note that the adjoint method requires only one evaluation of Φ at $\bar{x}$ to compute $\nabla J(\bar{x})$.

5.3 Evaluation of the draping function

The draping problem consists in finding one local minimum of the potential energy, which amounts to solving the following problem,

$$\text{Given } \bar{x}, \text{ find } x \quad \text{s. t.} \quad F(x, \bar{x}) = 0 \text{ and } \mathrm{D}_x F(x, \bar{x}) \succ 0. \tag{23}$$

Naive draping algorithm. As already mentioned, this problem has no unique solution, and thus one cannot define a true draping function, at least globally. Still, it is easy to find an admissible x by minimizing the function $x \mapsto E_p(x, \bar{x})$ locally (using for instance the Newton method), starting from the initial guess $x_0 = x_\mathrm{t}$. By analogy with the draping function, let us call Φ this draping algorithm, which takes as input $\bar{x}$ and returns x.

Unfortunately, even though it is initialized with x_t, such a method is likely to return a local minimum *that is far from* x_t. In this case, the inverse method for solving (20) may get stuck at a wrong place and thus hardly converge to a global minimum. Furthermore, our procedure Φ is not continuous w.r.t. the $\bar{x}$ variable, as two close positions for $\bar{x}$ might lead to completely different local minima x. Hence, during inversion it may happen that two subsequent steps of the adjoint evaluate Φ at two unrelated places. In this case the computation of ∇J becomes meaningless.

Regularized draping algorithm. Our goal is to build a draping procedure that remains *consistent* with our inversion algorithm. To this aim, we penalize the energy to be minimized so as to avoid falling into a local minimum which is far from the target x_t. That is, we consider the new potential energy

$$E_p^\lambda(x, \bar{x}) = E_p(x, \bar{x}) + \frac{\lambda}{2} \left\| x - x_\mathrm{t} \right\|^2, \tag{24}$$

where $\lambda \geq 0$ is a regularization factor. Choosing a high value for λ helps "convexify" the potential energy around x_t. Of course it also leads to an energy that is quasi-independent of $\bar{x}$, thus far from the original one. Conversely, setting a low value restores the original energy, but we loose the benefit of penalization. Note that if E_p reaches a minimum at x_t for a given $\bar{x}$, then it is also a minimum for E_p^λ, $\forall \lambda \geq 0$. This encourages us to decrease the value of λ over successive calls of Φ during inversion, when the confidence in $\bar{x}$ increases (see Algorithm 1).

Our new procedure Φ^λ thus consists in minimizing $x \mapsto E_p^\lambda(x, \bar{x})$, starting from $x_0 = x_\mathrm{t}$. In practice we use the Newton-CG method [Nocedal and Wright 2006, Section 6.2], to perform the minimization, that is we solve at each step k the following linear system in Δ_{k+1},

$$\mathrm{D}_x F^\lambda(x_k, \bar{x})\, \Delta_{k+1} = -F^\lambda(x_k, \bar{x}) \tag{25}$$

where $\Delta_{k+1} = x_{k+1} - x_k$ and $F^\lambda(x, \bar{x}) = \nabla_x E_p^\lambda$. Compared to the pure Newton algorithm, the Newton-CG method has the advantage of always finding a descent direction Δ_{k+1}, in particular in the case when $\mathrm{D}_x F(x_k, \bar{x})$ is not positive-definite. To accelerate convergence, we perform an adaptive increment along Δ_{k+1} using a Wolfe linesearch. Our draping algorithm is summarized in Algorithm 3.

For most of our examples, we obtained convergence in a few iterations only. This good convergence rate — which contrasts with some earlier feedbacks on the Newton method [Volino and Magnenat-Thalmann 2007] — is explained by the fact that minimization starts from the target x_t, which is already fairly close to the solution.

To simplify notations, we will omit the superscript λ and simply write E_p and F to refer to this regularized potential and the regularized internal forces respectively in the rest of the paper.

ALGORITHM 3: Draping procedure Φ

Data: Target equilibrium pose x_t, regularization factor λ, current natural pose $\bar{x}$
Result: $x = \Phi(\bar{x})$, a local minimum of the shell potential energy, this energy being evaluated at $\bar{x}$ and λ-regularized towards x_t.
Procedure `Draping`(x_t, λ, $\bar{x}$)
1 $x \leftarrow$ `Newton-CG`($E_p^\lambda(\bullet, \bar{x})$, $x_\mathrm{init} = x_\mathrm{t}$);
2 **return** x;

5.4 Removing degrees of freedom

In Step 1, we focus on the contact-free zones of the shell and we leave contact zones aside. To this end, we fix the vertices corresponding to contact points in the undeformed and deformed configurations as we will shortly describe, and we run a slightly modified version of the inversion algorithm. We use a similar procedure to handle the points in the model that are meant to be physically attached, such as the points located near the centerline of the book in the page example shown in Figure 1.

We assume that the set of contact vertices in the deformed shape, whose set of indices is denoted by I_c, is known a priori and that their positions match the ones of the corresponding target vertices. This is justified by the fact that in the case of a successful inversion, the target shape and the optimized draped shape should (almost) coincide. To handle these fixed points, we draw inspiration from the technique used by Baraff and Witkin [1998] to prevent certain vertices from moving in certain space directions using a filtering process, equivalent to an orthogonal projection of the equations [Ascher and Boxerman 2003].

Let Π be the orthogonal projection matrix applied to the system, defined as

$$[\Pi x]_i = \begin{cases} 0 & \text{if i is the index of a fixed vertex} \\ x_i & \text{otherwise,} \end{cases} \tag{26}$$

where $x_i \in \mathbb{R}^3$ denotes here the coordinates of the point i of x. Π can be used to 'remove' the unnecessary degrees of freedom from the system in order to build a smaller system as described below.

Let n_c be the number of fixed points and $\hat{\Pi} : \mathbb{R}^{3n} \to \mathbb{R}^{3(n-n_c)}$ the operator that removes the fixed components. We denote by $\hat{\Pi}^\top$ the "inverse" operator that maps a vector from $\mathbb{R}^{3(n-n_c)}$ back to $\mathbb{R}^{3n}$ by adding zeros to the vector at the right locations. We note that we have $\hat{\Pi}^\top \circ \hat{\Pi} = \Pi$. The system can then be parameterized by the

"true" degrees of freedom $\hat{x} \in \mathbb{R}^{3(n-n_c)}$ and $\hat{\bar{x}} \in \mathbb{R}^{3(n-n_c)}$ that allow us to reconstruct the variables

$$x = \hat{\Pi}^\top \hat{x} + (I - \Pi)x_\mathrm{t} \in \mathbb{R}^{3n} \tag{27}$$

$$\bar{x} = \hat{\Pi}^\top \hat{\bar{x}} + (I - \Pi)x_\mathrm{t} \in \mathbb{R}^{3n}. \tag{28}$$

This lets us work with the new energy

$$\hat{E}_p : (\hat{x}, \hat{\bar{x}}) \mapsto E_p(\hat{\Pi}^\top \hat{x} + (I - \Pi)x_\mathrm{t}, \hat{\Pi}^\top \hat{\bar{x}} + (I - \Pi)x_\mathrm{t}), \tag{29}$$

whose gradient $\nabla_{\hat{x}} \hat{E}_p = -\hat{F}$ with respect to $\hat{x}$ is

$$\nabla_{\hat{x}} \hat{E}_p = \hat{\Pi} \nabla_x E_p(\hat{\Pi}^\top \hat{x} + (I - \Pi)x_\mathrm{t}, \hat{\Pi}^\top \hat{\bar{x}} + (I - \Pi)x_\mathrm{t})). \tag{30}$$

Finally, since we have $\|\Pi^\top \hat{x} + (I - \Pi)x_\mathrm{t} - x_\mathrm{t}\| = \|\hat{x} - \hat{\Pi} x_\mathrm{t}\|$, the problem we aim to solve can be written as

$$\min_{\substack{(\hat{x}, \hat{\bar{x}}) \\ \hat{F}(\hat{x}, \hat{\bar{x}})=0 \\ \nabla^2_{\hat{x}} \hat{E}_p(\hat{x}, \hat{\bar{x}})>0}} \frac{1}{2}\|\hat{x} - \Pi x_\mathrm{t}\|^2. \tag{31}$$

Noting that $D_{\hat{x}} \hat{F} = \hat{\Pi} \circ D_x F \circ \hat{\Pi}^\top$ and $D_{\bar{x}} \hat{F} = \hat{\Pi} \circ D_{\bar{x}} F$, System (22) becomes

$$\begin{cases} [D_{\hat{x}} \hat{F}]^\top \hat{p} = \hat{\Pi}(x_\mathrm{t} - x) \\ \Pi \nabla J(\bar{x}) = \Pi [D_{\bar{x}} \hat{F}]^\top \hat{p} \end{cases} \iff \begin{cases} \hat{\Pi}[D_x F]^\top \Pi p = \hat{\Pi}(x_\mathrm{t} - x) \\ \Pi \nabla J(\bar{x}) = \Pi [D_{\bar{x}} F]^\top \Pi p \end{cases}$$
$$\iff \begin{cases} \Pi D_x F]^\top \Pi p = \Pi(x_\mathrm{t} - x) \\ \Pi \nabla J(\bar{x}) = \Pi [D_{\bar{x}} F]^\top \Pi p \end{cases} \tag{32}$$

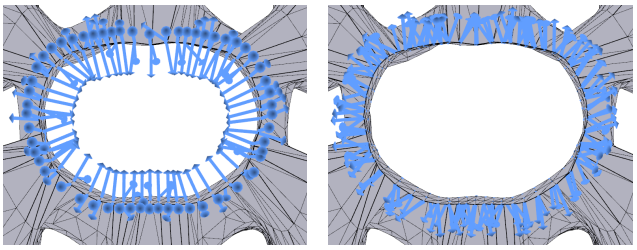
where p satisfies $\hat{\Pi} p = \hat{p}$.

To solve System (32) in practice, we first solve the adjoint equation using the filtering method by Baraff and Witkin [1998] and then we evaluate the gradient of J using the so-computed adjoint state (Πp). Note that the second equation of the system is projected on the image of Π only because we ask the components of $\bar{x}$ corresponding to fixed vertices in x to be fixed as well.

6 ACCOUNTING FOR FRICTIONAL CONTACT

Since contact vertices are assumed to be fixed due to static friction, it might be tempting to stop the algorithm after Step 1. However, fixing points is not sufficient to deal with true frictional contact, as this can yield arbitrarily oriented forces. This issue is illustrated in Figure 5. It clearly shows that the resulting bilateral constraint forces, i.e., the forces required to fix the points (computed as $f_{b_i} = F(x_{\frac{1}{2}}, \bar{x}_{\frac{1}{2}})_i$ with $i \in I_c$), may be outside their respective (virtual) Coulomb friction cones or even be oriented towards the obstacle — corresponding in the latter case to artificial *adhesive* forces. The goal of Step 2 is precisely to rectify these forces, i.e., to bring them back inside the friction cones, by adjusting the natural pose adequately. We further illustrate the importance of Step 2 in our result section (see Section 8.3).

We do not restrict the optimization to the set of contact points, since a change in their positions might affect the equilibrium of nearby contact-free points. Instead we tackle the problem globally, and adjust the entire natural mesh.



(a) Inversion with fixed points. (b) Inversion with frictional contact.

Fig. 5. Comparison of the yielded interaction forces in the bilateral case (left) and in the unilateral case with dry friction (right) at the waist of **Gored skirt** (top view, normalized vectors).

6.1 Admissible set of frictional contact forces

Given an initial $(x_{\frac{1}{2}}, \bar{x}_{\frac{1}{2}})$ pair satisfying static equilibrium with bilateral constraints, our idea is to enforce the frictional contact constraints by projecting the natural pose $\bar{x}_{\frac{1}{2}}$ onto a manifold $G_K \circ F = 0$ where G_K, that we will shortly define, satisfies

$$y \in K \iff G_K(y) = 0. \tag{33}$$

We assume that G_K acts on a per component basis, i.e.,

$$[G_K(y)]_i = G_{K_i}(y_i), \quad y \in \mathbb{R}^{3n}, \quad i = 1..n.$$

Then, a canonical choice for the indices corresponding to contact-free points, i.e., the ones for which $K_i = \{0\}$, is the identity operator

$$G_{K_i} = I, \quad i \notin I_c,$$

with I as differential. In contrast, when $i \in I_c$, the differential of G_{K_i} will always be null in the interior of $K_i = K(e_i, \mu_i)$ regardless of the specific choice for G_{K_i}, as long as the latter satisfies Condition (33).

Let $g : \mathbb{R} \to \mathbb{R}_+$ be a function satisfying

$$g(t) \begin{cases} > 0 & \text{if } t < 0 \\ = 0 & \text{otherwise.} \end{cases}$$

In practice, we choose $g(t) = -t$ on $\mathbb{R}^-$. We then set $G_{K_i} = h_i$ for $i \in I_c$, where $h_i : \mathbb{R}^3 \to \mathbb{R}$ is given by

$$h_i(u) = g(u_N) + g(\mu_i u_N - \|u_T\|),$$

with $u_N = u^\top \frac{e_i}{\|e_i\|}$ and $u_T = u - u_N \frac{e_i}{\|e_i\|}$. It is easy to check that

$$\forall u \in \mathbb{R}^3 \quad h_i(u) = 0 \iff (u_N \geq 0 \text{ and } \mu_i u_N \geq \|u_T\|)$$
$$\iff u \in K(e_i, \mu_i).$$

6.2 Relaxing the projection onto the admissible set

Having a suitable function G_K at hand, we do not solve an exact projection problem, which would be too cumbersome. Instead we solve for the following minimization problem,

$$\min_{\bar{x}} \frac{1}{2}\|G_K(F(x_{\frac{1}{2}}, \bar{x}))\|^2, \tag{34}$$

which moves the forces as close as possible to the admissible set. In practice our algorithm always reaches a minimum value very close to zero, meaning that our resulting forces properly satisfy frictional contact constraints.

Still, we shall point out that due to the high nonlinearity of G_K and F, convergence to $G_K \circ F = 0$ is not granted a priori. Therefore, properly warm-starting the algorithm using $\bar{x}_{\frac{1}{2}}$ is key to the success of the approach (see below). From a practical point of view, we minimize Problem (34) using a BFGS algorithm with backtracking line search, which relies on the computation of the gradient of G_K. Although this gradient is discontinuous at certain points, we did not encounter any issue in practice. We note that it is possible to build a smooth function G_K by using the alternative function $h_i : u \mapsto g(u_N) + g(\mu_i^2 u_N^2 - \|u_T\|^2)$ where g is here defined by $g(t) = t^2$ if $t < 0$ and $g(t) = 0$ otherwise. However we found in our experiments that the presence of higher degree terms in this formulation damped the convergence of the algorithm when approaching the solution.

Finally, we have seen earlier that Step 2 was mandatory to get a consistent set of frictional contact forces. Conversely, Step 1 is key to the success of our global algorithm. Indeed, reaching $G_K(F((\bar{x}))) = 0$ is equivalent to having a $\bar{x}$ that yields a static equilibrium under frictional contact forces. However, this formulation alone provides no guarantee on the *stability* of the equilibrium. Yet, our goal after the inversion is to perform a consistent simulation, in which the computed configuration cannot drift away from the target after a tiny perturbation. This justifies the use of Step 1, which guarantees the finding of a "stable" equilibrium, thanks to the call to the draping function. The importance of Step 1 for warm-starting Step 2 is further illustrated in Section 8.3.

7 IMPLEMENTATION DETAILS

7.1 Algorithm parameters

We use the regularized potential energy (24) in both Step 1 and Step 2 of our algorithm. In practice, we arbitrarily start with $\lambda = 1000$ and we divide λ by 2 after each step of the outer loop of the algorithm. To ensure that no non-physical force remains, we set λ to 0 when its value is below the threshold $\epsilon_\lambda = 0.1$, as reported in Algorithm 1.

Step 1 stops when $\|\delta\bar{x}\|_\infty \leq 1.e{-}15$ or $\|\nabla\Phi(\bar{x})\|_\infty \leq 1.e{-}15$, or when 2000 iterations are reached. Step 2 stops when $\|\delta\bar{x}\|_\infty \leq 1.e{-}15$ or $\|\nabla G(\bar{x})\|_\infty \leq 1.e{-}15$, or when 5000 iterations are reached.

7.2 Inclusion in the convex set K

Constraining a variable to lie in an open set K as defined by equation 15 is numerically hazardous. Such a strict enforcing is however important since the frontier between sticking and sliding behaviors precisely occurs when contact forces move from the interior to the boundary of the friction cone. We use a simple trick to enforce this strict set-membership and ensure that the shell will not move at resimulation, even if some numerical error (e.g., number truncation) occurs: we define a slightly lower friction coefficient $\mu_i' = \beta\mu_i$ (in practice we chose $\beta = 0.975$), and consider K_i as the *closed* set

$$K_i = \begin{cases} C(e_i, \mu_i') & \text{if } i \in I_c \\ \{0\} & \text{otherwise.} \end{cases} \tag{35}$$

7.3 Machine architecture

Our method was implemented in C/C++ with a single-threaded architecture, on a PC featuring 4 dual-core Intel i7-5600U processors running at 2.60GHz.

8 RESULTS

8.1 Framework

We have run our method on nine different examples which are listed in Table 1, together with the parameters used for inversion and further animation. These target shapes represent various common objects made of a thin elastic surface, such as a paper page, garments, or accessories like soft hats — all of them requiring contact and friction with an external object to be posed, for instance a human head or body. While some of our examples replicate the shape of real objects, others are deliberately stylized in a cartoon-like fashion to reflect the wide scope of our approach, which is not restricted to a specific kind of surface but can accommodate any arbitrary smooth mesh as input. Finally, our example data come from various sources, ranging from the output of a shell simulator to manual free-form design, and even automatic garment capture. The diversity in shapes and design modes successfully handled by our algorithm illustrates the versatility and robustness of our approach.

For each one of these examples, we first choose a set of material parameters. Mass density and stiffness parameters are set constant per example. Although not a limitation of our approach, for the sake of simplicity we also set a unique friction coefficient per example. Then, we apply our inversion algorithm and retrieve a natural shape $\bar{x}$. We finally run a direct dynamic simulation out of it, that is we set the natural shape of the simulator to $\bar{x}$, and we set the initial configuration of the shell to the target shape. This process is called *resimulation*. Resimulation is made on two purposes: first, to check the stable equilibrium of the target subject to gravity and contact constraints (the target should remain still); then, to further animate the target and see how it deforms under new external stresses induced for instance by wind forces or body motion. Below we present and analyze the results given by our inversion algorithm on all these examples, including resimulation, for different settings. Please also see our accompanying video for animated illustrations of our results.

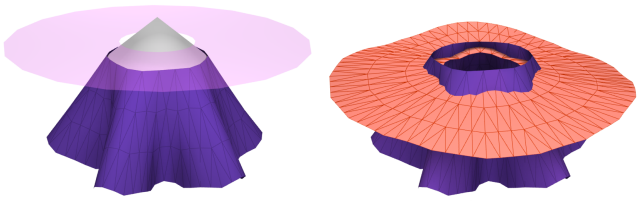
Table 1. Material parameters for our inversion examples.

Example	Source	ρ	k_L	k_θ	μ
Synthetic Skirt	Simulation	0.1	3.0	$5.0e^{-3}$	0.6
Book Page	3D Design	10.	5.0	$2.0e^{-3}$	0.4
Beret	3D Design	0.25	$5.0e^{-2}$	$5.0e^{-4}$	0.3
Floppy Hat	3D Design	2.5	5.0	$5.0e^{-4}$	0.6
Top	3D Design	0.5	0.2	$1.0e^{-1}$	0.2
Saroual	3D Design	0.05	1.0	$1.0e^{-2}$	0.2
Puff Sleeve	3D Design	1.0	0.1	$1.0e^{-3}$	0.7
Gored Skirt	3D Design	0.05	0.8	$5.0e^{-3}$	0.7
Clothcap Shirt	Capture	1.0	3.0	$5.0e^{-4}$	0.2

8.2 Qualitative results

Target generated by simulation. The **Synthetic Skirt** case is a toy example generated by the discrete shell dynamic simulator [Grinspun et al. 2003], using a flat torus as the natural shape and as initial state. To pose the object, gravity was applied and frictional contact activated between the torus and a conical obstacle positioned inside it. During simulation, the torus sags under gravity, sliding around

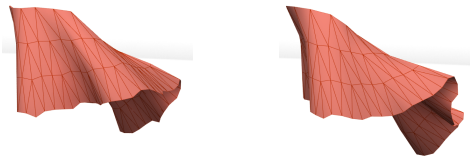
the cone and finally coming to rest when elastic and frictional contact forces balance gravity, taking the shape of a flared skirt. This equilibrium shape was registered as our target shape, which was in turn inverted using exactly the same material parameters as the ones used for the initial simulation.



(a) Target (b) Computed natural shape

Fig. 6. The **Synthetic Skirt** target (a - in purple) was generated by simulation from a flat torus (a - in semi-transparent pink). During inversion, the natural shape flattens and shrinks around the waist (b - in orange).

Inversion results are presented in Figure 6. As expected, throughout iterations the natural shape raises up, flattens, and shrinks around the waist to compensate for stretching and sliding. When the algorithm has converged, it is noteworthy that the resulting natural shape is drastically different from the target shape taken as the initial point of our inversion algorithm; this is in contrast to previous methods which are better suited for small, local changes of the natural shape. However, while our inversion algorithm perfectly converges to a high precision, the retrieved natural shape is not perfectly flat, and thus does not exactly match the original one. This is a good illustration of the non-uniqueness of our inverse problem. This point is further discussed in Section 9.3. Finally, if we depart from the reference material parameters chosen for direct simulation, inversion keeps on working well within a reasonable range of values. We have inverted **Synthetic Skirt** two more times, first by dividing the stretching stiffness by two, second by multiplying the bending stiffness by two. In the two cases the initial target shape is a stable equilibrium, but further motion trajectories under wind differ substantially (see Figure 7).



(a) Smaller stretching stiffness (b) Stronger bending stiffness

Fig. 7. Resimulation of **Synthetic Skirt** after inversion with different material parameters

Target manually designed. Out of our nine test cases, seven were freely designed by an artist, using the Autodesk 3ds max software for modeling the surface target and creating the animation of external contacting bodies for resimulation. For all these examples, the only available input we had was the mesh of the target produced by the artist. For inversion, we have set all material parameters (mass, bending and stretching stiffness, friction coefficients) so as to obtain some desired softness and roughness at contact during resimulation. Of course, this choice was conditional to the existence of a solution to our inverse problem. For instance, in the **Saroual** case, which represents cartoonish baggy pants (see Figure 8), only stiff enough materials can account for a static equilibrium. This point is discussed in Section 9.2.

The first example, **Book Page**, represents a page forming an arch on top of other pages of an open book (see Figure 1). The arch was modeled by pure intuition, relying on a cylinder patch, and without caring about any physical consideration. When simulating this target without inversion, the page slides and unfolds over the book. In contrast, after inversion with frictional contact, the page remains perfectly still during resimulation even though the material chosen is very soft. When applying weak wind forces, the page slides and sticks again, keeping its arch-like shape. If wind forces are strong enough, the page finally takes off and flips onto the other half of the book.

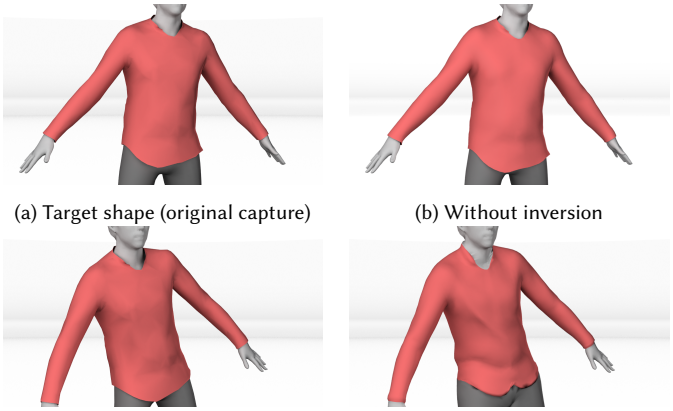
The two hat examples, namely **Beret** and **Floppy Hat**, are typical examples of accessories made as soft elastic shells, which are posed on a human head through contact and friction. Without inversion, **Floppy Hat** sags, completely losing its original style and partly covering the face of the character. In contrast, after our inversion, the hat preserves its original style while convincingly flip-flopping during fast rotation of the head (see Figure 2). For **Beret**, which is placed slanted on the side of the head, friction with the head is crucial to ensure it is not going to fall on the ground when subject to a small external perturbation. Without inversion, the resulting deflation of the beret, even though it remains small, is sufficient to yield a fatal fall of the object. After inversion however, the beret correctly remains inflated and posed on the head. If a light wind is applied, it slides on the head and eventually sticks a bit further. Of course, if stronger wind occurs, the beret cannot resist and flies away, as expected.

Finally, to show the versatility of our approach, we have considered four other cartoonish and fancy cloth examples, **Top & Saroual**, **Puff Sleeve**, and **Gored Skirt** (see Figure 8). Note that these garments would be particularly challenging to generate from pattern design, whereas 3D free-form design is a natural way for creating such shapes (see our supplementary video which shows a live design session for **Gored Skirt**, performed by our artist).

Target automatically reconstructed from capture. Automatic reconstruction of clothing from real human motion is now an active topic in computer graphics and vision. Very recently, Pons-Moll et al. [2017] have presented a technique which automatically extracts the deforming garment geometry from the analysis of a human body sequence filmed by cameras. Their output are two clean mesh sequences in time, one for the estimated nude body and one for the garment. Hence, a pair consisting of one garment mesh with the corresponding body mesh exactly matches the input format of our inversion algorithm, provided the pair is chosen at an instant where the character is static (which is generally the case at the beginning of the sequence). We have considered inverting a shirt captured with their system (namely **ClothCap Shirt**), which is posed on a

Fig. 8. Five cartoonish examples, namely **Beret**, **Top & Sarouel**, **Puff Sleeve**, and **GoredSkirt**, inverted by our method and consistently animated. Note that in **Puff Sleeve**, the sleeve is not attached to the dress and properly fits around the arm only due to friction.

male body (see Figure 9 (a)). Even though the contacting regions between the garment and the body meshes do not have as much precision as our freely designed examples, our method still converges to a plausible natural shape, and prevents sagging at resimulation. Moreover, the shirt can then be further simulated in a convincing manner on the reconstructed body mesh sequence, using our own material parameters (see Figure 9 (c)). In Figure 9 (d) we show the actual result of capture during animation. Of course, the shape is different from the one generated by our simulation, as we did not try to fit our material parameters to match the capture. We leave this interesting research topic for future work (see Section 9.4).



(a) Target shape (original capture) (b) Without inversion

(c) Simulation after inversion (d) Reference (original capture)

Fig. 9. Inverse design of a real shirt, **ClothCap Shirt**, captured and extracted from the body by Pons-Moll and colleagues [2017].

8.3 Evaluation

Measure of deformation. Figure 10 shows the relative stretching strain and the bending strain, respectively, of the computed natural shape compared to the target. The relative stretching strain is a signed relative value, computed as $\frac{\bar{L}_e - L_e}{L_e}$. When negative (blue color), it means that the natural shape has been compressed compared to the target. When positive (red color), it means that it has been dilated. Non stretching is represented with the white color. The bending strain is a non-negative value, computed as $\|\bar{\theta}_e - \theta_e\|$. It is color-coded with the red color which is all the more intense as the value is large.

A first observation is that our inversion algorithm is able to locally stretch, compress and/or bend the target shape so as to compute a final natural shape satisfying the desired properties. Moreover, although compression is predominant on our examples, dilatation still plays an important role, especially for the beret and puff sleeve examples where the natural shape needs to be inflated to account for the target. Note that such examples could not have been treated with methods limited to compression, such as [Bartle et al. 2016].

Role of frictional contact during inversion (Step 2). Our inversion algorithm first considers that points are fixed at contact (Step 1), before adjusting the forces so that they satisfy Coulomb constraints (Step 2). One may wonder why the second step is absolutely necessary, as approximating contacts as fixed points during inversion could, at first sight, been thought of as a viable and cheaper solution.

In order to show that the second step is not an option, we have compared two inversion results, **Book Page** and **Gored Skirt**, against their respective counterparts inverted using fixed points (i.e., restricting our algorithm to the first step only). After inversion, if simulation is performed using fixed points again, then the target

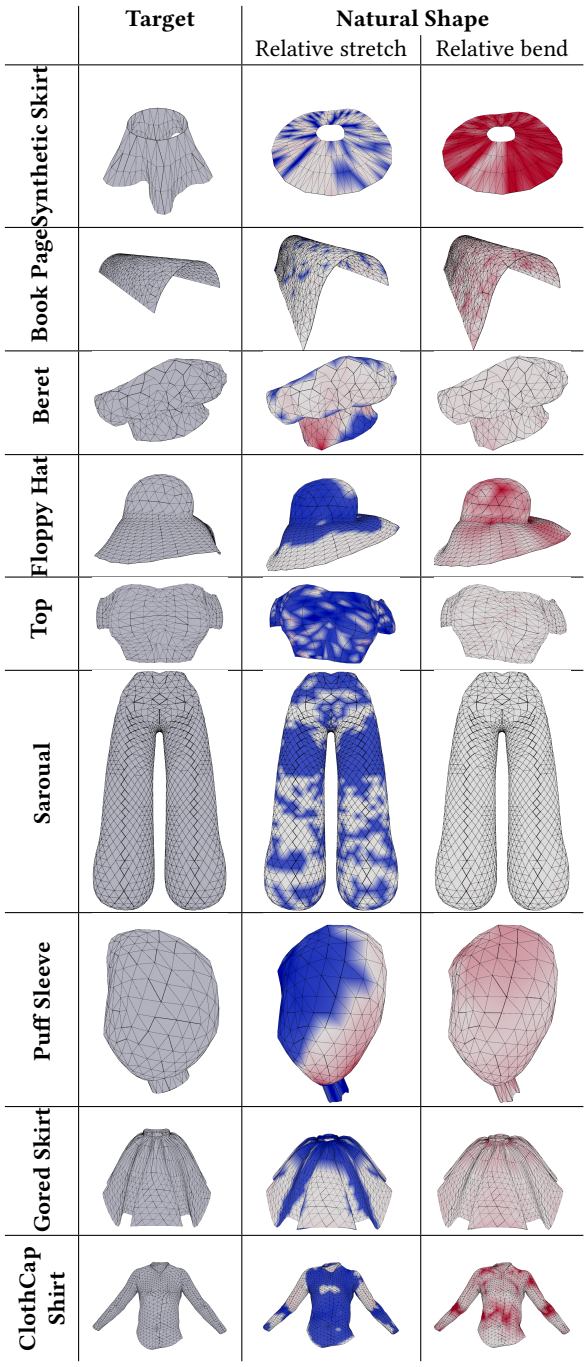


Fig. 10. For each of our examples, we display the target (first column) and the natural shape found by our algorithm (second and third columns) based on the material parameters described in Table 1. Relative stretching and bending of the natural shape w.r.t the target are depicted in color code in the second and third columns, respectively. Note that on some examples like **Beret**, **Puff Sleeve**, or **ClothCap Shirt**, not only local compression but also local dilatation occurs on the natural shape.

shape remains at equilibrium as expected. However, during resimulation, motion of the object may look very unrealistic, as parts of the object vertices are forced to remain fixed; this is depicted in Figure 11 on **Book Page** subject to external wind forces. Now, to have a more natural motion, one may think of simulating the inverted object using frictional contact. Unfortunately, if the friction coefficient μ chosen is not strong enough, equilibrium will not be satisfied and the object will sag under gravity. For some cases like **Book Page**, where each bilateral force corresponding to fixed point inversion happens to lie in the half space spanned by the contact normal, then choosing a strong enough friction coefficient might be sufficient to guarantee equilibrium. This however requires manual parameter tweaking, and does not give to the user the freedom to choose the friction coefficient she desires. Even more annoying, in the general case a strong friction coefficient won't make it either, since the bilateral forces computed from inversion may be oriented downwards the obstacle, thus corresponding to *adhesive* forces. This is typically the case of points located at the waist of skirts or pants, see the example of **Gored Skirt** in Figure 5a. In such cases sagging will irremediably occur, whatever the friction coefficient chosen.

In contrast, after including Step 2 in our algorithm, the solution found is guaranteed to match equilibrium for any friction coefficient.The only restriction comes from the quality of convergence of our algorithm, which depends on the choice of parameters. This dependency is analyzed below and further discussed in Section 9.2.

Role of the inversion with the draping function (Step 1). Conversely, one could then question the importance of Step 1. Indeed, reaching the objective $G_K(F^\lambda(\bar{x})) = 0$ in Step 2 means that the conditions of a static equilibrium are fulfilled. However, the static case is not our sole concern, as the computed couple $(x, \bar{x})$ shape is intended to be used for further animation. Yet, if the x found is an unstable equilibrium, any small perturbation (some noise at the beginning of the resimulation for instance) can shift it to another stable configuration, therefore ruining our efforts to preserve the target. In contrast, in the adjoint method used in Step 1, we have seen that the free points of the equilibrium are in a stable configuration thanks to the draping procedure. Initializing Step 2 with the result of Step 1 enables to perform a correction only at the points in contact while propagating this correction along the rest of the shell. The free points, starting from a stable equilibrium, are very unlikely to be shifted to an unstable equilibrium, and in practice we never met such a case.

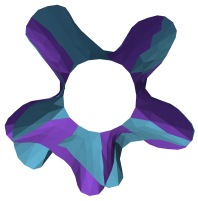


Fig. 12. Target shape (purple), stable equilibrium (blue)

To illustrate that Step 1 is key to stability, **Synthetic skirt** has been inverted using Step 2 only. The solver actually converges to a solution $\bar{x}$ such that the target is at equilibrium. However, the introduction of a small perturbation (a variation of magnitude $1.e-2$ in the value of the gravity) makes the target shift to another equilibrium (see Figure 12), while the same simulation using the output of our full algorithm (Step 1 + Step 2) is stable.

(a) Target shape (b) Sagging after incomplete inversion (c) Resimulation under wind using fixed points

Fig. 11. Skipping frictional contact when inverting the book page (a). If one performs incomplete inversion by replacing frictional contact forces with fixed points (step 1 only), then sagging still occurs at resimulation (b). The target remains at stable equilibrium only if fixed points are used again for resimulation; however subsequent motion under wind effect appears unrealistic as the page is neither able to slide nor to take off (c).

8.4 Convergence and performance

Detailed performance and convergence results of our algorithm on all our simulation examples are provided in Table 2.

An important result is that our inversion algorithm converges well for all our examples, while applied onto a range of different material parameters. However, the choice of the material parameters may influence the quality of convergence. On one end of the spectrum, the larger the material parameters are (stiff and highly frictional material), the easier the inverse problem is. At the limit of rigidity, the inverse solution is the target shape itself. Of course, in such cases, further animation of the object is not interesting. On the other end of the spectrum, that is, for a soft and slippery material, for which a richer dynamics can be expected, there is a high risk that the inverse problem has no solution. In such a case our method fails to converge. In practice, when this happened (typically for **Saroual** or **Gored Skirt**), we simply chose a stiffer material and rerun the inversion. However such a manual iterative process can become tedious, and in future work (see Section 9.2) we would like to make it much simpler and automatic.

Then, it is noteworthy that Step 1 of the algorithm is always at least as costly as Step 2, and often appears as the main bottleneck of our approach. Likewise, one obvious observation is that the minimization problems in both Step 1 and Step 2 are affected by the dimension of the problem. As expected, overall the higher the number of vertices is, the slower the convergence rate is. Avenues to reduce the overall computational cost and scale up our algorithm to larger meshes are suggested in Section 9.1.

9 LIMITATIONS AND FUTURE WORK

We have presented the first algorithm to invert deformable shells subject to both gravitational and dry frictional contact forces. Our approach is robust and effective in terms of approximation quality, which we have demonstrated on a wide range of examples. Nevertheless, our system has some limitations and opens many exciting opportunities for future work.

9.1 Performance

Currently, the biggest limitation of our system is its performance, as computing the natural shape can reach several hours or even days for the largest examples. In practice, most of the computational cost comes from the use of the draping function, which needs to be evaluated multiple times during each step of the adjoint method. While our computational timings might seem large compared to alternative approaches that only aim at satisfying force balance, this is the price to pay to guarantee that the equilibrium that we have found is actually stable. In a different context, Wang [2018] has recently proposed some interesting techniques to improve the efficiency of his draping function and gradient-based optimizer. We plan to draw inspiration from his work to improve the performance of our algorithm.

9.2 Choice of material parameters

We currently leave the choice to the user for setting the garment material properties. In our view, this degree of freedom left to the user appears beneficial in the context of an animation creation process. Indeed, while a designer cannot easily guess the natural pose of a deformed configuration, she generally has some idea about the kind of material (silk, linen) she would like to simulate. Moreover, similarly to some studies in fiber design [Derouet-Jourdan et al. 2010], we have checked in practice that for a continuous range of the material properties, any given shell configuration can be interpreted as a stable equilibrium. This means that within this range, the designer cannot choose a "wrong" material, and our method will always converge to a natural pose which enables to interpret the input shape as a stable equilibrium configuration. The main problem in the case of shells is that evaluating the bounds of this range is extremely difficult. In practice, if the user chooses a too soft material, there may not exist a solution. In this case the solver will fail to converge, and the user will have to choose a stiffer material before running inversion again. This manual process is of course all the most cumbersome as each inversion may take several hours to be performed. As future work, we would like to investigate systematic techniques to characterize, given a target mesh, the range of material parameters for which a valid solution can be generated. This would allow us to provide the user with guaranteed bounds on the set of feasible parameters *a priori*, as was done in the case of isolated fibers [Derouet-Jourdan et al. 2010].

9.3 Towards garment inversion

Being focused on animation purposes, we intentionally relied on a shell model to explain the target geometry, which offers more expressiveness compared to a fixed set of flat patterns. This allowed

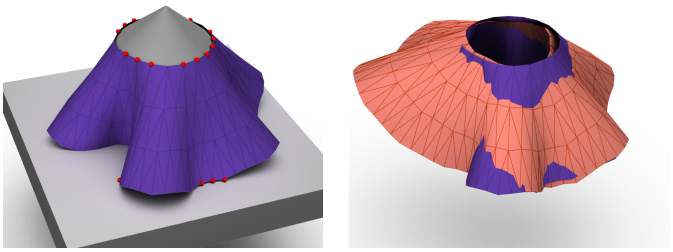
Table 2. Performance of our inversion algorithm for all our examples

Example	n_v [a]	n_c [a]	$t^1_{exec}(\%)$ [b]	$\bar{n}^1_{iter}$ [c]	$t^2_{exec}(\%)$ [d]	$\bar{n}^2_{iter}$ [e]	$t_{exec}(min)$ [f]	$\Phi(\bar{x})$ [g]	$G(\bar{x})$ [g]
Synthetic Skirt	200	19	75	598	25	157	4	0	$1.5e^{-23}$
Book Page	817	143	46	75	54	2338	187	$1.5e^{-11}$	$1.2e^{-23}$
Puff Sleeve	402	67	68	509	32	389	14	0	$7.4e^{-23}$
Beret	846	69	48	883	52	3055	97	0	$4.5e^{-25}$
Soft Hat	876	178	51	500	49	723	50	0	$2.3e^{-22}$
Top	900	116	82	472	18	1800	382	$1.4e^{-19}$	$1.3e^{-19}$
Saroual	2500	61	26	858	74	3432	950	0.	$4.5e^{-22}$
Gored Skirt	1255	168	47	1115	53	3149	427	$6.1e^{-22}$	$7.1e^{-22}$
ClothCap Shirt	2550	931	20	633	80	2751	750	0	$4.0e^{-23}$

[a] Number of vertices (n_P) and contact points (n_c)
[b] Percentage of the computation time spent in Step 1
[c] Average number of iterations per λ in Step 1
[d] Percentage of the computation time spent in Step 2
[e] Average number of iterations per λ in Step 2
[f] Total computation time
[g] Final error $\Phi(\bar{x})$ and constraint error $G(\bar{x})$

us to handle arbitrary target meshes, such as the cylindrical **Book Page**, for which no flat inverse solution exists. We could also tackle a number of fancy garment examples with a particular design, such as **Gored Skirt**, which are more naturally modeled as stiff shells. In practice, we have not observed any discrepancy when resimulating our inverted examples, even when the retrieved natural shape appeared more curved than what expected, such as in the **Synthetic Skirt** example. Indeed, in such a case the retrieved natural bending angles remain small compared to the deformed bending angles, hence the new bending energy remains close to the original one. This property may however no longer hold when dealing with soft garments subject to folds and wrinkles due to contact, as explained below.

In most of the garments we have been considering for inversion, roundish shapes and sharp folds are intended by the artist as part of the design itself, and do not stem from contact interactions. Instead, contacts are always localized at tight parts of the mesh (e.g., at the belt). In contrast, when contact appears at arbitrary locations of the mesh, the inverse solution which is found is not necessarily the most appropriate among the large set of feasible solutions. Indeed, because the first step of our algorithm fixes all the contacting points, a preferred solution for the natural shape is the one that best sticks to the target and replicates the same folding patterns. This phenomenon can be observed on a modified version of **Synthetic Skirt**, which is now touching the ground (see Figure 13). During inversion, the natural shape is not able to raise up anymore, and another solution, closer to the target but less intuitive, is found. This problem actually arises whenever one wants to invert a soft tissue in contact with the body, such as the **Arabesque Dress** mesh depicted in Figure 14(a), which results from realistic cloth simulation [Li et al. 2018a]. We have run our inversion solver on this specific mesh, using high stretching and low bending parameters. Our solver has properly converged in a few hours, however the solution found is far from being satisfying, as shown in Figure 14(b). During resimulation, thin folds fail to vanish when contact ceases, thus resulting into unrealistic animation.



(a) Target shape (b) Computed natural shape

Fig. 13. Our approach may suffer from an inappropriate selection of the natural shape among all possible solutions. The test case here is similar to the **Synthetic Skirt** one (Figure 6) except that a ground has been added. Apart from a few contact points with the ground (marked by the red spheres), the simulated equilibrium is close to the original one (a - in purple). However, the resulting computed natural shape (b - in orange) flattens less as these points are fixed during Step 1.

The latter case is a typical example where one would rather like to have the tissue flatten during inversion. This request is even stronger if the final application is to fabricate real cloth from flat patterns: in such a case the recovered natural shape must be made of developable patches. It would be interesting in the future to investigate how to guide the solver and favor locally developable natural shapes, so as to yield more consistent inverse solutions in the case of soft garments. A more consistent solution avoiding stick-to-contact natural shapes would also be desirable to properly handle self-contact, which we currently ignore. Finally, a long-term direction of research would

(a) Target shape (b) Computed natural shape

Fig. 14. The natural shape (b) computed from the target pose (a - output from [Li et al. 2018a]) exhibits undesirable folds.

be to identify automatically flat patterns from our guided inverse solution, so as to be able to fabricate arbitrary 3D designs in real.

9.4 Towards parameter identification from real data

We have shown with **ClothCap Shirt** that our algorithm performed well on complex geometries captured from real data, even in the absence of specific knowledge regarding the material parameters of the real garment. A natural avenue for future work would be to extend our approach to material parameter fitting, e.g., by leveraging multiple captured poses. The resulting geometry could then be used either to identify real material parameters, or to improve the output of the captured system, by offering a consistent and realistic guide to the reconstruction algorithm.

ACKNOWLEDGMENTS

We would like to thank Gilles Daviet, Nicolas Pautet and Charles Dapogny for early discussions about this work. We are also grateful to Gerard Pons-Moll and his co-authors for sharing with us some of their recent garment capture data. Many thanks to Raphaël Charrondière for his help with creating the supplementary video. Finally, we would like to thank the anonymous reviewers for their useful comments. This work was supported in part by the ERC grant GEM (StG-2014-639139).

REFERENCES

V. Acary and B. Brogliato. 2008. *Numerical methods for nonsmooth dynamical systems.* Lecture Notes in Computational and Applied Mechanics, Vol. 35. Springer.

U. Ascher and E. Boxerman. 2003. On the modified conjugate gradient method in cloth simulation. *The Visual Computer* 19, 7-8 (2003), 526–531.

D. Baraff and A. Witkin. 1998. Large Steps in Cloth Simulation. In *Computer Graphics Proceedings (Proc. ACM SIGGRAPH'98)*. 43–54.

A. Bartle, A. Sheffer, V. Kim, D. Kaufman, N. Vining, and F. Berthouzoz. 2016. Physics-driven Pattern Adjustment for Direct 3D Garment Editing. *ACM Transactions on Graphics* 35, 4, Article 50 (July 2016), 11 pages. https://doi.org/10.1145/2897824.2925896

J. Beck and K. Woodbury. 1998. Inverse problems and parameter estimation: integration of measurements and analysis. *Measurement Science and Technology* 9, 6 (1998), 839.

D. Bradley, T. Boubekeur, and W. Heidrich. 2008. Accurate multiview reconstruction using robust binocular stereo and surface meshing. In *Computer Vision and Pattern Recognition (CVPR'08)*.

R. Brouet, A. Sheffer, L. Boissieux, and M.-P. Cani. 2012. Design Preserving Garment Transfer. *ACM Transactions on Graphics* 31, 4 (2012), 36:1–36:11.

M. Carignan, Y. Yang, N. Magnenat-Thalmann, and D. Thalmann. 1992. Dressing Animated Synthetic Actors with Complex Deformable Clothes. *SIGGRAPH Comput. Graph.* 26, 2 (July 1992), 99–104. https://doi.org/10.1145/142920.134017

R. Casati, G. Daviet, and F. Bertails-Descoubes. 2016. *Inverse elastic cloth design with contact and friction.* Research Report. Inria Grenoble, Université de Grenoble. https://hal.archives-ouvertes.fr/hal-01309617

X. Chen, C. Zheng, W. Xu, and K. Zhou. 2014. An Asymptotic Numerical Method for Inverse Elastic Shape Design. *ACM Transactions on Graphics* 33, 4, Article 95 (July 2014), 11 pages. https://doi.org/10.1145/2601097.2601189

G. Daviet, F. Bertails-Descoubes, and L. Boissieux. 2011. A hybrid iterative solver for robustly capturing Coulomb friction in hair dynamics. *ACM Transactions on Graphics* 30 (2011), 139:1–139:12. Issue 6.

G. Daviet, F. Bertails-Descoubes, and R. Casati. 2015. Fast Cloth Simulation with Implicit Contact and Exact Coulomb Friction. ACM SIGGRAPH / Eurographics Symposium on Computer Animation. (Aug. 2015). https://hal.inria.fr/hal-01180756 Poster.

A. Derouet-Jourdan, F. Bertails-Descoubes, G. Daviet, and J. Thollot. 2013. Inverse Dynamic Hair Modeling with Frictional Contact. *ACM Transactions on Graphics* 32, 6, Article 159 (Nov. 2013), 10 pages. https://doi.org/10.1145/2508363.2508398

A. Derouet-Jourdan, F. Bertails-Descoubes, and J. Thollot. 2010. Stable Inverse Dynamic Curves. *ACM Transactions on Graphics* 29, Article 137 (December 2010), 10 pages. Issue 6. https://doi.org/10.1145/1882261.1866159

M. Giles and N. Pierce. 2000. An Introduction to the Adjoint Approach to Design. *Flow, Turbulence and Combustion* 65, 3-4 (2000), 393–415. https://doi.org/10.1023/A:1011430410075

E. Grinspun, A. Hirani, M. Desbrun, and P. Schröoder. 2003. Discrete Shells. In *ACM SIGGRAPH - EG Symposium on Computer Animation (SCA'03)*. ACM-EG SCA, 62–67.

S. Hadap. 2006. Oriented strands - dynamics of stiff multi-body system. In *ACM SIGGRAPH - EG Symp. on Comp. Anim. (SCA'06)*. ACM-EG SCA, 91–100.

T. Igarashi and J. Hughes. 2003. Clothing Manipulation. *ACM Transactions on Graphics* 22, 3 (July 2003), 697–697. https://doi.org/10.1145/882262.882328

C. Li, H. Pan, Y. Liu, X. Tong, A. Sheffer, and W. Wang. 2017. BendSketch: Modeling Freeform Surfaces Through 2D Sketching. *ACM Transactions on Graphics* 36, 4, Article 125 (July 2017), 14 pages. https://doi.org/10.1145/3072959.3073632

J. Li, G. Daviet, R. Narain, F. Bertails-Descoubes, M. Overby, G. Brown, and L. Boissieux. 2018a. An Implicit Frictional Contact Solver for Adaptive Cloth Simulation. *ACM Transactions on Graphics* 37, 4, Article 52 (Aug. 2018), 15 pages.

M. Li, A. Sheffer, E. Grinspun, and N. Vining. 2018b. FoldSketch: Enriching garments with physically reproducible folds. *ACM Transactions on Graphics* 37, 4, Article 133 (Aug. 2018), 13 pages.

Marvelous Designer. 2010. http://www.marvelousdesigner.com. (2010).

J. Nocedal and S. Wright. 2006. *Numerical Optimization.* Springer.

G. Pons-Moll, S. Pujades, S. Hu, and M. Black. 2017. ClothCap: Seamless 4D Clothing Capture and Retargeting. *ACM Transactions on Graphics* 36, 4, Article 73 (July 2017), 15 pages. https://doi.org/10.1145/3072959.3073711

S. Porumbescu, B. Budge, L. Feng, and K. Joy. 2005. Shell Maps. *ACM Transactions on Graphics* 24, 3 (July 2005), 626–633. https://doi.org/10.1145/1073204.1073239

M. Skouras, B. Thomaszewski, B. Bickel, and M. Gross. 2012. Computational Design of Rubber Balloons. *Computer Graphics Forum (Proc. Eurographics)* (2012).

M. Skouras, B. Thomaszewski, P. Kaufmann, A. Garg, B. Bickel, E. Grinspun, and M. Gross. 2014. Designing Inflatable Structures. *ACM Transactions on Graphics* 33, 4, Article 63 (July 2014), 10 pages. https://doi.org/10.1145/2601097.2601166

E. Turquin, J. Wither, L. Boissieux, M.-P. Cani, and J. Hughes. 2007. A Sketch-Based Interface for Clothing Virtual Characters. *IEEE Comput. Graph. Appl.* 27, 1 (Jan. 2007), 72–81. https://doi.org/10.1109/MCG.2007.1

C. Twigg and Z. Kačić-Alesić. 2011. Optimization for sag-free simulations. In *ACM SIGGRAPH - EG Symposium on Computer Animation (SCA'11)*. ACM-EG SCA, 225–236. https://doi.org/10.1145/2019406.2019437

N. Umetani, D. Kaufman, T. Igarashi, and E. Grinspun. 2011. Sensitive Couture for Interactive Garment Editing and Modeling. *ACM Transactions on Graphics* 30, 4 (2011). http://www.cs.columbia.edu/cg/SC/

P. Volino, F. Cordier, and N. Magnenat-Thalmann. 2005. From early virtual garment simulation to interactive fashion design. *Computer-Aided Design Journal (CAD journal)* 37 (March 2005), 593–608.

P. Volino and N. Magnenat-Thalmann. 2007. Stop-and-Go Cloth Draping. *The Visual Computer* (August 2007), 669–677.

B. Wang, L. Wu, K. Yin, U. Ascher, L. Liu, and H. Huang. 2015. Deformation Capture and Modeling of Soft Objects. *ACM Transactions on Graphics* 34, 4, Article 94 (July 2015), 12 pages. https://doi.org/10.1145/2766911

H. Wang. 2018. Rule-free sewing pattern adjustement with precision and efficiency. *ACM Transactions on Graphics* 37, 4, Article 53 (Aug. 2018), 14 pages.

R. White, K. Crane, and D. Forsyth. 2007. Capturing and animating occluded cloth. *ACM Transactions on Graphics* 26, 3, Article 34 (July 2007). https://doi.org/10.1145/1276377.1276420